Address Book Framework

Session 012

Address Book Framework

John Geleynse
User Experience Evangelist

Address Book Framework

Henri Lamiraux
Engineering Manager

AddressBook.framework

- Long-standing request from developers
- Accessible from Carbon, Cocoa, and AppleScript
- Thread Safe
- Concurrent access by multiple applications
- Extensible Property Set

AddressBook.framework

- Centralized storage for contacts
- Heavily used in Jaguar
  - Mail
  - iChat
  - Address Book
  - Sherlock
  - Setup Assistant
  - Etc . . .

AddressBook.framework

- Jaguar and later only (not backward compatible)
- Automatically import and convert Mac OS X 10.1 database

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

ABRecord

- Row in Address Book Database
- Unique ID
- Abstract base class for:
    - ABPerson
    - ABGroup
- “Dictionary” of Property/Value pairs

ABRecord

```
-(NSString *)uniqueId;

-(BOOL)setValue:(id)value
    forProperty:(NSString*)property;

-(id)valueForProperty:(NSString *)property;

-(BOOL)removeValueForProperty:(NSString
  *)property;
```

ABRecord

```
-(NSString *)uniqueId;

-(BOOL)setValue:(id)value
    forProperty:(NSString*)property;

-(id)valueForProperty:(NSString *)property;

-(BOOL)removeValueForProperty:(NSString
  *)property;
```

ABRecord

```
ABRecord *record = …;

id value = [record valueForProperty:kSomeProperty];

if (value == nil) {
   <Do something>
} else {
   <Do something else>
}

[record setValue:value forProperty: kSomeProperty];
```

ABRecord

```
ABRecord *record = …;

id value = [record valueForProperty:kSomeProperty];

if (value == nil) {
   <Do something>
} else {
   <Do something else>
}

[record setValue:value forProperty: kSomeProperty];
```

ABRecord

```
ABRecord *record = …;

id value = [record valueForProperty:kSomeProperty];

if (value == nil) {
   <Do something>
} else {
   <Do something else>
}

[record setValue:value forProperty: kSomeProperty];
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

Properties

- Unique Name
  - FirstName, LastName, etc . . .
- Type
  - Integer, string, date, . . .
- Two flavors
  - Single and Multi-values

Properties

- Single Value Types

  kABStringProperty (NSString)
  kABIntegerProperty (NSNumber)
  kABRealProperty (NSNumber)
  kABDateProperty (NSDate)
  kABArrayProperty (NSArray)
  kABDictionaryProperty (NSDictionary)
  kABDataProperty (NSData)

Properties

- Examples of Single Value Property
  - First Name—kABStringProperty
  - Last Name—kABStringProperty
  - Birthday—kABDateProperty

Properties

- Single Values don't work for emails, phone numbers, etc . . .
- Need to be able to associate multi-labeled values with one property:

  Home: **jdoe@mac.com**

  Home: **jdoe@earthlink.net**

  Work: **jdoe@apple.com**

Properties

- Multi-value Property types

  kABMultiStringProperty
  kABMultiIntegerProperty
  kABMultiRealProperty
  kABMultiDateProperty
  kABMultiArrayProperty
  kABMultiDictionaryProperty
  kABMultiDataProperty

- ABMultiValue class encapsulates a multi-value

ABMultiValue

```
ABPerson *person = …;

NSString *firstName =
  [person valueForProperty:kABFirstNameProperty];

ABMultiValue *emails =
  [person valueForProperty:kABEmailProperty];
```

ABMultiValue

```
ABPerson *person = …;

NSString *firstName =
   [person valueForProperty:kABFirstNameProperty];

ABMultiValue *emails =
   [person valueForProperty:kABEmailProperty];
```

ABMultiValue

- Collection class like NSDictionary or NSArray
- Manages an “array” of triplets
    - Label
    - Value
    - Identifier

ABMultiValue

- Labels do not have to be unique
  - E.g., multiple “home” email addresses

ABMultiValue

- All values must have the same type
  - E.g., kABMultiStringProperty -> all NSString
  - E.g., kABMultiDateProperty -> all NSDate

ABMultiValue

- Identifier provides a reference to a specific value
  - Do not save away an index

ABMultiValue

- Notion of a “primary” value
  - Joe’s primary email is **joe@mac.com** (home)

ABMultiValue

- Two flavors
  - ABMultiValue
  - ABMutableMultiValue

ABMultiValue

- Two flavors
  - ABMultiValue
  - ABMutableMultiValue

ABMultiValue

```
-(unsigned int)count;

-(int)indexForIdentifier:(NSString *)identifier;

-(id)valueAtIndex:(int)index;
-(NSString *)labelAtIndex:(int)index;
-(NSString *)identifierAtIndex:(int)index;

-(NSString *)primaryIdentifier;

-(ABPropertyType)propertyType;
```

ABMultiValue

```
-(unsigned int)count;

-(int)indexForIdentifier:(NSString *)identifier;

-(id)valueAtIndex:(int)index;
-(NSString *)labelAtIndex:(int)index;
-(NSString *)identifierAtIndex:(int)index;

-(NSString *)primaryIdentifier;

-(ABPropertyType)propertyType;
```

ABMultiValue

```
-(unsigned int)count;

-(int)indexForIdentifier:(NSString *)identifier;

-(id)valueAtIndex:(int)index;
-(NSString *)labelAtIndex:(int)index;
-(NSString *)identifierAtIndex:(int)index;

-(NSString *)primaryIdentifier;

-(ABPropertyType)propertyType;
```

ABMultiValue

```
-(unsigned int)count;

-(int)indexForIdentifier:(NSString *)identifier;

-(id)valueAtIndex:(int)index;
-(NSString *)labelAtIndex:(int)index;
-(NSString *)identifierAtIndex:(int)index;

-(NSString *)primaryIdentifier;

-(ABPropertyType)propertyType;
```

ABMultiValue

```
-(unsigned int)count;

-(int)indexForIdentifier:(NSString *)identifier;

-(id)valueAtIndex:(int)index;
-(NSString *)labelAtIndex:(int)index;
-(NSString *)identifierAtIndex:(int)index;

-(NSString *)primaryIdentifier;

-(ABPropertyType)propertyType;
```

ABMultiValue Sample Code

- Display all email addresses of a person

ABMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
  NSLog(@”label: %@ email: %@”,
                [emails labelAtIndex:i]
                [emails valueAtIndex:i]);
}
```

ABMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
  NSLog(@”label: %@ email: %@”,
                [emails labelAtIndex:i]
                [emails valueAtIndex:i]);
}
```

ABMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
  NSLog(@”label: %@ email: %@”,
                [emails labelAtIndex:i]
                [emails valueAtIndex:i]);
}
```

ABMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
  NSLog(@”label: %@ email: %@”,
                [emails labelAtIndex:i]
                [emails valueAtIndex:i]);
}
```

ABMultiValue

- Two flavors
  - ABMultiValue
  - ABMutableMultiValue

ABMutableMultiValue

```
-(NSString*)addValue:(id)value
           withLabel:(NSString*)label;

-(NSString*)insertValue:(id)value
              withLabel:(NSString*)label
                atIndex:(int)index;

-(BOOL)removeValueAndLabelAtIndex:(int)index;

-(BOOL)replaceValueAtIndex:(int)index
                 withValue:(id)value;
-(BOOL)replaceLabelAtIndex:(int)index
                 withLabel:(NSString*)label;

-(BOOL)setPrimaryIdentifier:(NSString*)identifier;
```

ABMutableMultiValue

```
-(NSString*)addValue:(id)value
           withLabel:(NSString*)label;

-(NSString*)insertValue:(id)value
              withLabel:(NSString*)label
                atIndex:(int)index;

-(BOOL)removeValueAndLabelAtIndex:(int)index;

-(BOOL)replaceValueAtIndex:(int)index
                 withValue:(id)value;
-(BOOL)replaceLabelAtIndex:(int)index
                 withLabel:(NSString*)label;

-(BOOL)setPrimaryIdentifier:(NSString*)identifier;
```

ABMutableMultiValue

```
-(NSString*)addValue:(id)value
           withLabel:(NSString*)label;

-(NSString*)insertValue:(id)value
              withLabel:(NSString*)label
                atIndex:(int)index;

-(BOOL)removeValueAndLabelAtIndex:(int)index;

-(BOOL)replaceValueAtIndex:(int)index
                 withValue:(id)value;
-(BOOL)replaceLabelAtIndex:(int)index
                 withLabel:(NSString*)label;

-(BOOL)setPrimaryIdentifier:(NSString*)identifier;
```

ABMutableMultiValue

```
-(NSString*)addValue:(id)value
           withLabel:(NSString*)label;

-(NSString*)insertValue:(id)value
              withLabel:(NSString*)label
                atIndex:(int)index;

-(BOOL)removeValueAndLabelAtIndex:(int)index;

-(BOOL)replaceValueAtIndex:(int)index
                 withValue:(id)value;
-(BOOL)replaceLabelAtIndex:(int)index
                 withLabel:(NSString*)label;

-(BOOL)setPrimaryIdentifier:(NSString*)identifier;
```

ABMutableMultiValue

```
-(NSString*)addValue:(id)value
           withLabel:(NSString*)label;

-(NSString*)insertValue:(id)value
              withLabel:(NSString*)label
                atIndex:(int)index;

-(BOOL)removeValueAndLabelAtIndex:(int)index;

-(BOOL)replaceValueAtIndex:(int)index
                 withValue:(id)value;
-(BOOL)replaceLabelAtIndex:(int)index
                 withLabel:(NSString*)label;

-(BOOL)setPrimaryIdentifier:(NSString*)identifier;
```

ABMutableMultiValue Sample Code

- Add an email address to a person

ABMutableMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
   valueForProperty:kABEmailProperty];

ABMutableMultiValue *
   mutableEmails = [emails mutableCopy];

[mutableEmails addValue:@”jdoe@apple.com”
                      withLabel:kABHomeEmailLabel];

[person setvalue:mutableEmails forProperty:
   kABEmailProperty];

[mutableEmails release];
```

ABMutableMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
    valueForProperty:kABEmailProperty];

ABMutableMultiValue *
    mutableEmails = [emails mutableCopy];

[mutableEmails addValue:@”jdoe@apple.com”
                      withLabel:kABHomeEmailLabel];

[person setvalue:mutableEmails forProperty:
    kABEmailProperty];

[mutableEmails release];
```

ABMutableMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
    valueForProperty:kABEmailProperty];

ABMutableMultiValue *
    mutableEmails = [emails mutableCopy];

[mutableEmails addValue:@”jdoe@apple.com”
                      withLabel:kABHomeEmailLabel];

[person setvalue:mutableEmails forProperty:
    kABEmailProperty];

[mutableEmails release];
```

ABMutableMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
    valueForProperty:kABEmailProperty];

ABMutableMultiValue *
    mutableEmails = [emails mutableCopy];

[mutableEmails addValue:@”jdoe@apple.com”
                      withLabel:kABHomeEmailLabel];

[person setvalue:mutableEmails forProperty:
    kABEmailProperty];

[mutableEmails release];
```

ABMutableMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
   valueForProperty:kABEmailProperty];

ABMutableMultiValue *
   mutableEmails = [emails mutableCopy];

[mutableEmails addValue:@”jdoe@apple.com”
                      withLabel:kABHomeEmailLabel];

[person setvalue:mutableEmails forProperty:
   kABEmailProperty];

[mutableEmails release];
```

ABMutableMultiValue Sample Code

```
ABPerson *person = …;

ABMultiValue *emails = [person
   valueForProperty:kABEmailProperty];

ABMutableMultiValue *
   mutableEmails = [emails mutableCopy];

[mutableEmails addValue:@”jdoe@apple.com”
                      withLabel:kABHomeEmailLabel];

[person setvalue:mutableEmails forProperty:
   kABEmailProperty];

[mutableEmails release];
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

ABPerson

- Subclass of ABRecord
- Represents a person

ABPerson

- Some Standard Properties

kABFirstNameProperty (kABStringProperty)
kABLastNameProperty (kABStringProperty)
kABBirthdayProperty (kABDateProperty)
kABEmailProperty (kABMultiStringProperty)
kABPhoneProperty (kABMultiStringProperty)
kABAddressProperty (kABMultiDictionaryProperty)

...

ABPerson

- Inherits ABRecord APIs
- Also provides

```
-(BOOL)setTIFFImageFromData:(NSData*)data;
-(NSData*)TIFFImageData;

-(NSArray*)parentGroups;
```

ABPerson

- Inherits ABRecord APIs
- Also provides

```
-(BOOL)setTIFFImageFromData:(NSData*)data;
-(NSData*)TIFFImageData;

-(NSArray*)parentGroups;
```

ABPerson vCard Support

- Can import and export vCards (3.0)

```
-(id)initWithVCardRepresentation:(NSData*)vCardData;
-(NSData*)vCardRepresentation;
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

ABGroup

- Subclass of ABRecord
- Represents a group of people and/or other groups
    - No circular references allowed

ABGroup

- One Standard Property

kABGroupNameProperty (kABStringProperty)

ABGroup

- Inherits ABRecord APIs
- Also provides:

```
- (NSArray*)members;
- (BOOL)addMember:(ABPerson*)person;
- (BOOL)removeMember:(ABPerson*)person;

- (NSArray*)subgroups;
- (BOOL)addSubgroup:(ABGroup*)group;
- (BOOL)removeSubgroup:(ABGroup*)group;

- (NSArray*)parentGroups;
```

ABGroup

- Inherits ABRecord APIs
- Also provides:

```
- (NSArray*)members;
- (BOOL)addMember:(ABPerson*)person;
- (BOOL)removeMember:(ABPerson*)person;

- (NSArray*)subgroups;
- (BOOL)addSubgroup:(ABGroup*)group;
- (BOOL)removeSubgroup:(ABGroup*)group;

- (NSArray*)parentGroups;
```

ABGroup

- Inherits ABRecord APIs
- Also provides:

```
- (NSArray*)members;
- (BOOL)addMember:(ABPerson*)person;
- (BOOL)removeMember:(ABPerson*)person;

- (NSArray*)subgroups;
- (BOOL)addSubgroup:(ABGroup*)group;
- (BOOL)removeSubgroup:(ABGroup*)group;

- (NSArray*)parentGroups;
```

ABGroup Distribution List

- For MultiValue properties only
- Most useful for Email Addresses
    - Allows groups to be used as a Mailing lists

      E.g., When sending a email to my group named “Friends” I want to use John’s home email but Paul’s work email . . .

ABGroup Distribution List

```
-(BOOL)setDistributionIdentifier:(NSString*)identifier
                     forProperty:(NSString*)property
                          person:(ABPerson*)person;

-(NSString*)distributionIdentifierForProperty:(NSString
   *)property person:(ABPerson*)person;
```

ABGroup Distribution List

```
-(BOOL)setDistributionIdentifier:(NSString*)identifier
                     forProperty:(NSString*)property
                          person:(ABPerson*)person;

-(NSString*)distributionIdentifierForProperty:(NSString
   *)property person:(ABPerson*)person;
```

ABGroup Distribution List

```
-(BOOL)setDistributionIdentifier:(NSString*)identifier
                     forProperty:(NSString*)property
                          person:(ABPerson*)person;

-(NSString*)distributionIdentifierForProperty:(NSString
   *)property person:(ABPerson*)person;
```

ABGroup Distribution List

```
-(BOOL)setDistributionIdentifier:(NSString*)identifier
                     forProperty:(NSString*)property
                          person:(ABPerson*)person;

-(NSString*)distributionIdentifierForProperty:(NSString
   *)property person:(ABPerson*)person;
```

ABGroup Distribution List

```
-(BOOL)setDistributionIdentifier:(NSString*)identifier
                     forProperty:(NSString*)property
                          person:(ABPerson*)person;

-(NSString*)distributionIdentifierForProperty:(NSString
   *)property person:(ABPerson*)person;
```

ABGroup Distribution List

```
-(BOOL)setDistributionIdentifier:(NSString*)identifier
                     forProperty:(NSString*)property
                          person:(ABPerson*)person;

-(NSString*)distributionIdentifierForProperty:(NSString
   *)property person:(ABPerson*)person;
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

ABAddressBook

- Represents the Address Book database

  ~/Library/Application Support/AddressBook

- Unique Shared Instance

ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;

- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;
- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;

- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;
- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;

- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;
- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook

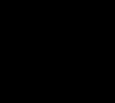
```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;

- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;
- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;

- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;
- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook Sample Code

- Add a person named “John Doe”



ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[[ABPerson alloc] init] autorelease];

[newPerson setValue:@”John”
        forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
        forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[[ABPerson alloc] init] autorelease];

[newPerson setValue:@”John”
        forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
        forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[[ABPerson alloc] init] autorelease];

[newPerson setValue:@”John”
        forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
        forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[[ABPerson alloc] init] autorelease];

[newPerson setValue:@”John”
        forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
        forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[[ABPerson alloc] init] autorelease];

[newPerson setValue:@”John”
        forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
        forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[[ABPerson alloc] init] autorelease];

[newPerson setValue:@”John”
        forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
        forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];

[addressBook save];
```

ABAddressBook Sample Code

- Create a group named “Friends” with two people: person1 and person2

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook Sample Code

```
ABAddressBook *addressBook;
ABGroup *newGroup;

addressBook = [ABAddressBook sharedAddressBook];

newGroup = [[[ABGroup alloc] init] autorelease];

[newGroup setValue:@”Friends”
        forProperty:kABGroupNameProperty];

[newGroup addPerson:person1];
[newGroup addPerson:person2];

[addressBook addRecord:newGroup];

[addressBook save];
```

ABAddressBook

- Searching

```
-(NSArray*)recordsMatchingSearchElement:(ABSearchElement
    *)search;
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

ABSearchElement

- Building blocks for queries
- Example:
  - Find all people who live in New York and have a mac.com email address
  - Need three search elements
    - S1 = “Home City = New York”
    - S2 = “Email contains mac.com”
    - S3 = S1 and S2

ABSearchElement

- Class method on ABPerson and ABGroup

```
+(ABSearchElement*)
        searchElementForProperty:(NSString*)property
                           label:(NSString*)label
                             key:(NSString*)key
                           value:(id)value
                      comparison:(ABSearchComparison)comparison;
```

ABSearchElement

- Search for all people that have “John” as first name

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@"John"
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@"John"
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement

- Search for all people living in “New York”

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement Sample Code

```
...

element = [ABPerson searchElementForProperty:kABAddressProperty
                                        label:kABHomeLabel
                                          key:kABAddressCityKey
                                        value:@"New York"
                                   comparison:kABEqualCaseInsensitive];

result = [addressBook recordsMatchingSearchElement:element];
```

ABSearchElement

- More complex search

```
+ (ABSearchElement*)
    searchElementForConjunction:(ABSearchConjunction)conjunction
                       children:(NSArray *)children;
```

- Conjunction

  kABSearchAnd
  kABSearchOr

ABSearchElement

- Search for all people that live in New York and have a mac.com email address

ABSearchElement Sample Code

```
ABAddressBook *addressBook;
ABSearcElement *element1, *element2, *element3;
NSArray *children;
NSArray *result;

addressBook = [ABAddressBook sharedAddressBook];
```

ABSearchElement Sample Code

```
element1 = [ABPerson
   searchElementForProperty:kABAddressProperty
                      label:kABHomeAddressLabel
                        key:kABAddressCityKey
                      value:@”New York”
                 comparison:kABEqualCaseInsensitive];
```

ABSearchElement Sample Code

```
element1 = [ABPerson
   searchElementForProperty:kABAddressProperty
                      label:kABHomeAddressLabel
                        key:kABAddressCityKey
                      value:@”New York”
                 comparison:kABEqualCaseInsensitive];
```

ABSearchElement Sample Code

```
element1 = [ABPerson
   searchElementForProperty:kABAddressProperty
                      label:kABHomeAddressLabel
                        key:kABAddressCityKey
                      value:@”New York”
                 comparison:kABEqualCaseInsensitive];
```

ABSearchElement Sample Code

```
element1 = [ABPerson
   searchElementForProperty:kABAddressProperty
                      label:kABHomeAddressLabel
                        key:kABAddressCityKey
                      value:@”New York”
                 comparison:kABEqualCaseInsensitive];
```

ABSearchElement Sample Code

```
element1 = [ABPerson
   searchElementForProperty:kABAddressProperty
                      label:kABHomeAddressLabel
                        key:kABAddressCityKey
                      value:@"New York"
                 comparison:kABEqualCaseInsensitive];
```

ABSearchElement Sample Code

```
element1 = [ABPerson
   searchElementForProperty:kABAddressProperty
                      label:kABHomeAddressLabel
                        key:kABAddressCityKey
                      value:@”New York”
                 comparison:kABEqualCaseInsensitive];
```

ABSearchElement Sample Code

```
element2 = [ABPerson
   searchElementForProperty:kABEmailProperty
                      label:nil
                        key:nil
                      value:@”mac.com”
   comparison:kABContainsSubStringCaseInsensitive];
```

ABSearchElement Sample Code

```
element2 = [ABPerson
    searchElementForProperty:kABEmailProperty
                       label:nil
                         key:nil
                       value:@”mac.com”
    comparison:kABContainsSubStringCaseInsensitive];
```

ABSearchElement Sample Code

```
element2 = [ABPerson
   searchElementForProperty:kABEmailProperty
                      label:nil
                        key:nil
                      value:@”mac.com”
   comparison:kABContainsSubStringCaseInsensitive];
```

ABSearchElement Sample Code

```
element2 = [ABPerson
    searchElementForProperty:kABEmailProperty
                       label:nil
                         key:nil
                       value:@”mac.com”
    comparison:kABContainsSubStringCaseInsensitive];
```

ABSearchElement Sample Code

```
element2 = [ABPerson
    searchElementForProperty:kABEmailProperty
                       label:nil
                         key:nil
                       value:@"mac.com"
    comparison:kABContainsSubStringCaseInsensitive];
```

ABSearchElement Sample Code

```
element2 = [ABPerson
    searchElementForProperty:kABEmailProperty
                       label:nil
                         key:nil
                       value:@”mac.com”
    comparison:kABContainsSubStringCaseInsensitive];
```

ABSearchElement Sample Code

```
children = [NSArray arrayWithObjects:element1,
                                     element2,
                                     nil];

element3 = [ABSearchElement
        searchElementForConjunction:kABSearchAnd
                           children:children];

result = [addressBook
                 recordsMatchingSearchElement:element3];
```

ABSearchElement Sample Code

```
children = [NSArray arrayWithObjects:element1,
                                     element2,
                                     nil];

element3 = [ABSearchElement
        searchElementForConjunction:kABSearchAnd
                           children:children];

result = [addressBook
                 recordsMatchingSearchElement:element3];
```

ABSearchElement Sample Code

```
children = [NSArray arrayWithObjects:element1,
                                     element2,
                                     nil];

element3 = [ABSearchElement
        searchElementForConjunction:kABSearchAnd
                           children:children];

result = [addressBook
                 recordsMatchingSearchElement:element3];
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

Notifications

- Two Notifications
  - kABDatabaseChangedNotification
    - Change was made by this process
  - kABDatabaseChangedExternallyNotification
    - Change was made by another process

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

Extending Property Set

- Class methods on ABPerson and ABGroup
- Properties are added and removed in batch

```
+ (int)addPropertiesAndTypes:(NSDictionary *)properties;
+ (int)removeProperties:(NSArray *)properties;
+ (NSArray *)properties;
+ (ABPropertyType)typeOfProperty:(NSString*)property;
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

C-based APIs

- Same as Obj-C
- Classes are represented by opaque types
    - ABPersonRef, ABGroupRef, ABMultiValueRef, . . .
- Uses CF naming convention and style

C-based APIs

```
ABAddressBookRef
   addressBook = ABGetSharedAddressBook();

ABPersonRef person = ABPersonCreate();

ABRecordSetValue(person, CFSTR(”John”),
                         kABFirstNameProperty);
ABRecordSetValue(person, CFSTR(”Doe”),
                         kABLastNameProperty);

ABAddRecord(addressBook, person);

CFRelease(person);
```

C-based APIs

```
ABAddressBookRef
   addressBook = ABGetSharedAddressBook();

ABPersonRef person = ABPersonCreate();

ABRecordSetValue(person, CFSTR(”John”),
                         kABFirstNameProperty);
ABRecordSetValue(person, CFSTR(”Doe”),
                         kABLastNameProperty);

ABAddRecord(addressBook, person);

CFRelease(person);
```

C-based APIs

```
ABAddressBookRef
   addressBook = ABGetSharedAddressBook();

ABPersonRef person = ABPersonCreate();

ABRecordSetValue(person, CFSTR(”John”),
                         kABFirstNameProperty);
ABRecordSetValue(person, CFSTR(”Doe”),
                         kABLastNameProperty);

ABAddRecord(addressBook, person);

CFRelease(person);
```

C-based APIs

```
ABAddressBookRef
   addressBook = ABGetSharedAddressBook();

ABPersonRef person = ABPersonCreate();

ABRecordSetValue(person, CFSTR(”John”),
                         kABFirstNameProperty);
ABRecordSetValue(person, CFSTR(”Doe”),
                         kABLastNameProperty);

ABAddRecord(addressBook, person);

CFRelease(person);
```

C-based APIs

```
ABAddressBookRef
   addressBook = ABGetSharedAddressBook();

ABPersonRef person = ABPersonCreate();

ABRecordSetValue(person, CFSTR(”John”),
                         kABFirstNameProperty);
ABRecordSetValue(person, CFSTR(”Doe”),
                         kABLastNameProperty);

ABAddRecord(addressBook, person);

CFRelease(person);
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- Notifications
- Extending Property Set
- C-based APIs
- AppleScript

AppleScript

- Full access to all AddressBook.framework APIs
- Access through the Address Book application

```
tell application "Address Book"
    …
```

Sample Script

- List all emails

```
tell application "Address Book"
    set all_people to people
    repeat with a_person in all_people
            set all_emails to emails of a_person
            repeat with a_email in all_emails
                    log (value of a_email) as string
            end repeat
    end repeat
end tell
```

Demo

Address Book Application

In Conclusion

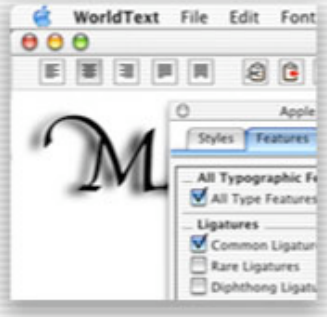
- Available in Jaguar
- Use AddressBook.framework
    - Important piece for a better user experience
    - Centralized information
    - Used extensively in Jaguar

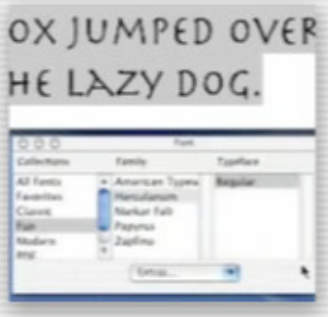
Who to Contact

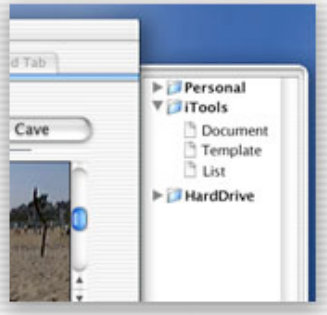
John Geleynse
User Experience Evangelist
Apple Worldwide Developer Relations
geleynse@apple.com

http://developer.apple.com/wwdc2002/urls.html

Q&A







John Geleynse
User Experience Evangelist
geleynse@apple.com

http://developer.apple.com/wwdc2002/urls.html


WWDC2002


WWDC2002


WWDC2002